REMARKS

In the Office Action, the Examiner indicated that claims 1-17 are pending in the application and the Examiner rejected all claims. The rejections are respectfully traversed below.

Summary of Rejections Based on Prior Art

On page 2 of the Office Action, the Examiner rejected claims 1-3 and 7-14 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,493,703 to Knight et al.

At page 6 of the Office Action, the Examiner rejected claims 4-6 and 15-17 under 35 U.S.C. §103(a) as being unpatentable over Kinght et al. in view of U.S. Patent No. 6,256,773 to Bowman-Anuah.

The Present Invention

The present invention is an asset locator (search engine) for locating software assets, code assets and the like that are stored in code repositories used by software designers. It provides the capability for the gathering of information about assets contained in the code repositories and the capturing of the gathered information in a database that can be used for the conducting of subsequent searches. The present invention has particular application in a software-development environment where the stored code assets may number in the millions and may be written in diverse languages such as, for example, Java, C/C++, COBOL, HTML, and/or XML.

A crawl process is performed on a storage device on which assets are stored to identify the assets. Asset-specific parameters related to the stored assets are identified, and the assets are then analyzed based upon these parameters. Textual and semantic information is extracted from the stored assets and then the extracted textual and semantic information is stored and indexed for retrieval.

In a preferred embodiment, a series of data analyzers that are specific to each type of data contained in the code repositories (e.g., a Java analyzer, a C/C++ analyzer, a COBOL analyzer, an HTML analyzer, and/or an XML analyzer) are integrated into the system so that they can be used to search the code repositories using particular attributes specific to the semantics of a particular language used to code the asset. In another preferred embodiment, the repositories are crawled automatically according to a schedule defined by the user, and the results of the crawling are stored in a database. Ordinary keyword searching can then be used with the system, either independently or combined with the attribute-specific semantic searching, to search the database.

U.S. Patent No. 6,493,703 to Knight et al.

U.S. Patent No. 6,493,703 to Knight et al. ("Knight") teaches an online message board system that monitors message traffic generated by subscribers so that intelligent decisions can be made concerning what types of content to locate and retrieve, what priority to use for locating such content, how to organize such content for ease of access by the subscribers, etc. A series of software robots are used to locate, retrieve and sort the content as derived from other online newsgroups. By studying subscriber message traffic, the system can self-tune itself automatically to constantly adjust content retrieval, storage and presentation in response to changing community interests, desires, and the like. In other words, Knight et al. teaches a system whereby message posts on a bulletin board system (BBS) are monitored and categorized based upon textual content of the posts.

U.S. Patent No. 6,256,773 to Bowman-Anuah

U.S. Patent No. 6,256,773 to Bowman-Anuah teaches a system, method and article of manufacture for affording consistency in a development architecture framework as components in the framework change. The Examiner relies upon Bowman-Anuah solely for the disclosure in column 3, line 50 to column 4, line 8; and column 8, line 60 to column 9, line 6, both of which are repeated herein in their entirety:

> "A preferred embodiment is written using JAVA, C, and the C++ language and utilizes object oriented programming methodology. Object oriented programming (OOP) has become increasingly used to develop complex applications. As OOP moves toward the mainstream of software design and development, various software solutions require adaptation to make use of the benefits of OOP. A need exists for these principles of OOP to be applied to a messaging interface of an electronic messaging system such that a set of OOP classes and objects for the messaging interface can be provided.
>
> OOP is a process of developing computer software using objects, including the steps of analyzing the problem, designing the system, and constructing the program. An object is a software package that contains both data and a collection of related structures and procedures. Since it contains both data and a collection of structures and procedures, it can be visualized as a self-sufficient component that does not require other additional structures. Procedures or data to perform its specific task. OOP, therefore, views a computer program as a collection of largely autonomous components, called objects, each of which is responsible for a specific task. This concept of

> packaging data, structures, and procedures together in one component or module is called encapsulation."
>
> * * * * *
>
> "Sun's (Java (language has emerged as an industry-recognized language for"programming the Internet." Sun defines Java as: "a simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded, dynamic, buzzword-compliant, general-purpose programming language. Java supports programming for the Internet in the form of platform-independent Java applets." Java applets are small, specialized applications that comply with Sun's Java Application Programming Interface (API) allowing developers to add "interactive content" to Web documents (e.g., simple animations, page adornments, basic games, etc.). Applets execute within a Java-compatible browser (e.g., Netscape Navigator) by copying code from the server to client."

The Examiner Has Not Set Forth a *Prima Facie* Case of Antipation

The MPEP and case law provide the following definition of anticipation for the purposes of 35 U.S.C. §102:

> "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." MPEP §2131 citing *Verdegaal Bros. v. Union Oil Company of California*, 814 F.2d 628, 631, 2 U.S.P.Q. 2d 1051, 1053 (Fed. Cir. 1987)

As set forth above, the present invention is directed to locating assets from within what is typically a voluminous storage of software assets and code assets in one or multiple code repositories. It is designed to function automatically and to search for semantic "cues" that allow more focused searches to be performed on the assets, beyond the searching typically available using strictly text-based searching. Specifically, code repositories are crawled to identify code assets stored therein, and data analyzers are provided for each type of data contained in the repository. Thus, if it is known that the code repositories contain Java assets, C++ assets, COBOL assets and HTML assets, then there will be separate analyzers for each type of data. The data in the repositories is "crawled" periodically and the asset specific analyzers identify <u>asset specific parameters</u> related to the stored assets and analyze them based upon these identified <u>asset specific parameters</u>. This enables the extraction of textual <u>and semantic</u> information from the stored assets, which are then stored and indexed for retrieval.

As an example, an organization might have 12 different code repositories containing assets numbering in the millions. Presume that a user wishes to locate Java programs that have utilized a hash table as the method attribute. If a prior art crawling system were used to locate any assets containing the term "hashtable" using existing text-based methods (such as those used by Knight), it is highly likely that many irrelevant documents would be retrieved, including documents which contain articles related to hash tables and the like. Using the present invention, since the data crawler has performed its process not only by text terms <u>but also by locating and indexing programs which have a method attribute</u> (a Java-specific parameter), all Java method attributes can be searched for the term "hashtable" in a query (as would be found in a Java program utilizing a hash table in a method attribute), and the user may now retrieve only assets which have this characteristic.

Knight et al. contains no teaching of the searching of code repositories looking for assets containing asset-specific semantical attributes and storing and retrieving them based upon these attributes. Knight et al. simply provides for a convenient way of categorizing posts on bulletin board systems, whereby the content of the posts is analyzed on an ongoing basis to adjust content retrieval, storage, and presentation. All of the data (posts) is of the same type, i.e., textual posts, i.e., there are no asset-specific semantical attributes and there is thus no searching performed based on asset-specific semantical attributes; all of the searching is text-based. While this may be beneficial to users of the bulletin board system, Knight et al. contains no teaching of identifying asset-specific parameters related to stored assets (as claimed herein), and thus cannot perform analyzing of stored assets based upon identified asset-specific parameters (as claimed herein). Further, Knight et al., while teaching the extraction of textual information, contains no teaching of the extraction of semantic information from the stored assets (as claimed herein), and thus cannot teach storing and indexing of semantic information for retrieval (as claimed herein). Each of these elements are specifically claimed in independent claims 1 and 7. Thus, each of the independent claims, and all claims depending therefrom, are patentable over Knight et al.

In addition to the above, the dependent claims add additional limitations not taught or suggested by Knight et al. For example, in claim 2, the stored assets comprise assets of diverse type, and the identifying step identifies the asset type of each stored asset. Claim 4 specifically requires that the stored assets comprise code assets and that the asset-specific parameters comprise languages in which each code asset is written. Claim 5 requires that the analysis that is performed using language-specific analyzers corresponding to the languages of said code assets. Further, claim 6 requires the analysis of the stored assets by the language-specific analyzers based upon predetermined parameters specific to the language to which they correspond.

Each of the elements claimed in the dependent claims further limit claim 1, and further distance the claims from the teachings of Knight. For example, nowhere in Knight is there a teaching or suggestion of searching code assets, nor is there teaching or suggestion of using language-specific analyzers, corresponding to languages of the code, for performing searches. Knight et al. simply teaches a method of text searching bulletin board posts and refining the categorization capability of the bulletin board posts based upon continued analysis of the subsequent posts. The dependent claims that depend from claim 7 contain similar limitations as those described with respect to claims 2-6 and thus define over Knight for the reasons set forth above. Accordingly, the Examiner is respectfully requested to reconsider and withdraw her rejection of claims 1-3 and 7-14 under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,493,703 to Knight et al.

<u>The Examiner has not Established a *prima facie* Case of Obviousness</u>

To support a rejection under 35 U.S.C. §103, a reason, suggestion, or motivation to lead an inventor to combine two or more references must be found. *Pro-Mold and Tool Co. v. Great Lakes Plastics Inc.*, 37 U.S.P.Q.2d 1627, 1629 (Fed.Cir. 1996) (see also MPEP 2143). The Examiner has not met this burden, as set forth below.

The Examiner asserts that claims 4-6 and 15-17 are unpatentable under 35 U.S.C. §103(a) based on Knight et al. in view of Bowman-Anuah.

The addition of the Bowman reference in combination with Knight does not render the claimed invention obvious. Like Knight, Bowman contains no teaching or suggestion of the identification of asset-specific parameters related to the stored assets; the analysis of the stored assets based on identified asset-specific parameters; the extraction of semantic information from the stored assets; and the storing and indexing of the extracted semantic information for retrieval. Since neither reference suggests, let alone teaches, those elements, if the test for a finding of obviousness has not been satisfied. Accordingly, the Examiner is respectfully requested to reconsider and withdraw her rejection of claims 4-6 and 15-17 under 35 U.S.C. §103(a) as being unpatentable of U.S. Patent No. 6,493,703 to Knight et al. in view of U.S. Patent No. 6,256,773 to Bowman-Anuah.

Conclusion

The rejection of claims 1-17 have been traversed. Accordingly, reconsideration of the present application, and withdrawal of the rejections on the grounds of 35 U.S.C. §§102 and 103 is respectfully requested.

A Petition for extending the time to respond to the Examiner's Action three months is enclosed in duplicate. The Commissioner is hereby authorized to charge any additional fees or credit any overpayment associated with this communication to Deposit Account No. 09-0461.

Respectfully submitted,

May 12, 2004
Date

Mark D. Simpson

Mark D. Simpson, Esq.
Registration No. 32,942
SYNNESTVEDT & LECHNER LLP
Suite 2600 Aramark Tower
1101 Market Street
Philadelphia, PA 19107
Telephone: (215) 923-4466
Facsimile: (215) 923-2189

M:\MSimpson\Clients\IBM Raleigh RSW\23546\patoff\Reply to action of 11122003.wpd